TIETO- JA SÄHKÖTEKNIIKAN TIEDEKUNTA

Markus Multamäki

DATAN ESIKÄSITTELY- JA VISUALISOINTITYÖKALU

Kandidaatintyö
Tietotekniikan tutkinto-ohjelma
Syyskuu 2020

Multamäki M. (2020) Datan esikäsittely- ja visualisointityökalu. Oulun yliopisto, tietotekniikan tutkinto-ohjelma. Kandidaatintyö, 31 s.

TIIVISTELMÄ

Koneoppimismenetelmien tehokkaaksi hyödyntämiseksi on tärkeää, että käyttäjällä on tietoa datajoukosta ja sen rakenteista. Tämän takia tavallinen ensimmäinen askel uuden datan tapauksessa on turvautua erilaisiin visualisointimenetelmiin. Visualisoinnin tarkoituksena on löytää samankaltaisuussuhteita datasta ja muodostaa alustavaa käsitystä sen rakenteista. Kiinnostavaa on esimerkiksi tietää datan jakautumisesta erilaisiin ryhmiin.

Ennen visualisointia moniulotteinen data on kuitenkin saatava pudotettua kahteen tai kolmeen ulottuvuuteen, jotta ihminen voisi tehdä siitä havaintoja. Tähän vastataan dimensionaalisuuden vähentämismenetelmillä. Dimensionaalisuuden vähentämisellä on visualisoinnin lisäksi roolinsa koneoppimisessa myös piirteiden tehostamisessa.

Dimensionaalisuuden aiheuttamien ongelmien lisäksi useimmat koneoppimismenetelmät vaativat datan skaalausta tai normalisointia ennen niiden käyttöä. Skaalaus tai normalisointi on yleisesti tärkeää, sillä useassa tapauksessa datan piirteiden arvoalueet poikkeavat toisistaan huomattavasti.

Tässä kandidaatintyössä on perehdytty datan skaalauksiin ja normalisointeihin, sekä dimensionaalisuuden vähentämiseen erilaisilla menetelmillä. Lisäksi on tutkittu erilaisten data-aineistojen rakennetta käsittelemällä niitä edellä mainituin keinoin. Työn tarkoituksena on valaista data-aineistoon perehtymisen tärkeyttä ja esitellä menetelmiä, joilla koneoppimisen tuloksia voidaan parantaa.

Työssä on kehitetty Python-kielinen työkalu, jonka avulla datan käsittely ja visualisointi onnistuu helposti graafisen käyttöliittymän avulla. Se on ensisijaisesti tarkoitettu opetustarkoituksiin.

Avainsanat: skaalaus, normalisointi, dimensionaalisuuden vähentäminen, koneoppiminen, data-analyysi

Multamäki M. (2020) Data preprocessing and visualization tool. University of Oulu, Degree Programme in Computer Science and Engineering. Bachelor's Thesis, 31 p.

ABSTRACT

For effective utilization of machine learning methods, it is important that the user has information about the dataset and its structures. Therefore, it is common to use visualization as the first step when dealing with new data. The purpose of visualization is to find similarities in data and to get insight about its structures. For example, it is interesting to find out whether the data is clustered.

Before visualization, the data needs to be transformed into two or three dimensions so that humans can make observations from it. This is the step where dimensionality reduction is used. In addition, dimensionality reduction plays role in machine learning when features are required to be more efficient.

On top of the problems caused by dimensionality, many machine learning methods require input data to be somehow scaled or normalized. Scaling or normalization is important because it is common that features in datasets are in different scales and distributions.

This bachelor's thesis introduces and experiments with different methods for data scaling, normalization and dimensionality reduction. Various real-life datasets and their structures are explored with these methods. The purpose of this is to underline the importance of gaining familiarity with new datasets and to introduce some common methods that can be used to improve results of machine learning methods.

The concrete contribution of this thesis is a data analysis tool developed using Python programming language. The tool is primarily intended for educational purposes and it makes data handling and visualization easier with the use of a graphical user interface.

Key words: data, scaling, normalization, dimensionality reduction, machine learning, data-analysis

SISÄLLYSLUETTELO

ALKULAUSE

Haluan kiittää ohjaajaani professori Olli Silvéniä erinomaisista neuvoista ja ohjauksesta työn tekemisen aikana. Hänen kommenteistaan oli huomattavaa apua työn tekemisessä. Kiitos myös tohtori Satu Tammiselle työn kommentoinnista.

Oulu, syyskuu 1. 2020

Markus Multamäki

LYHENTEIDEN JA MERKKIEN SELITYKSET

PCA	Pääkomponenttianalyysi
MDS	Monidimensioskaalaus
t-SNE	t-distributed stochastic neighbor embedding
UMAP	Uniform manifold approximation and projection
LVM	Latent variable model
k-NN	k-lähimmän naapurin luokittelija
SVM	Tukivektorikone

1. JOHDANTO

Datan skaalaus ja normalisointi sekä dimensioiden vähentäminen ovat säännöllisesti tärkeitä koneoppimis- ja data-analyysimenetelmien esikäsittelyvaiheissa. Syynä tähän on tarjolla olevan datan piirteiden vaihtelevat arvoalueet ja suuri dimensionaalisuus. Esimerkiksi kukin 256x256 pikselin harmaasävykuvan pikseleistä voidaan tulkita piirteeksi, jolloin kuvaa vastaavan piirrevektorin pituus on 65536. Jokainen piirre vastaa yhtä dimensiota. Datajoukon piirteet siis virittävät moniulotteisen piirreavaruuden, jossa jokainen piste vastaa tässä tapauksessa uniikkia kuvaa.

Suuri osa koneoppimismenetelmistä ajautuu ongelmiin syötedatan ollessa piirteiltään vaihteleva-arvoista ja dimensioltaan hyvin korkeaa. Esimerkiksi nykyisin hyvin suosituissa neuroverkoissa datan esikäsittelylle on tarvetta. Erityisesti vasta-alkajilla voi olla vaikeuksia saada tehokkaitakin koneoppimismenetelmiä toimimaan odotetulla tavalla johtuen puutteellisesta esikäsittely-ymmärryksestä.

Skaalaus ja normalisointi sekoitetaan usein toisiinsa, mutta niillä on merkittävä ero: skaalauksessa piirteet muunnetaan tietylle arvovälille, kun taas normalisointi vaikuttaa piirteen arvojakaumaan. Kaikella merkityksellisellä datalla on yleensä jokin rakenne, jonka löytymiseen ja visualisointiin ihmiselle esimerkiksi kaksi- tai kolmidimensioisena voidaan suuresti vaikuttaa skaalauksella ja normalisoinnilla. Toisaalta myös koneoppimismenetelmät oppivat datasta rakenteita, jolloin skaalauksella ja normalisoinnilla voi olla suuri vaikutus niiden suorituskykyyn.

Suuren piirteiden määrän aiheuttamiin ongelmiin vastataan usein dimensionaalisuuden vähentämismenetelmillä. Dimensionaalisuuden vähentämismenetelmät ovat kehittyneet vuosien saatossa lineaaristen menetelmien, kuten jo vuonna 1933 esitellyn pääkomponenttianalyysin [1] ja vuoden 1952 klassisen monidimensioskaalauksen [2] pohjalta epälineaarisiin menetelmiin. Eräs tällainen menetelmä on edellä mainittujen päälle rakennettu Isomap [3].

Epälineaaristen menetelmien kehittämiseen on johtanut perinteisten lineaaristen menetelmien haasteet epälineaarisia riippuvuussuhteita sisältävän datan dimensioiden pudotuksessa. Todelliset data-aineistot ovat usein jakautuneet moniulotteisille, epälineaarisille monistoille ja lineaarisilla menetelmillä tällaisen datan rakenteiden tunnistaminen ei onnistu kovin hyvin. Nykypäivänä suosittuja epälineaarisia menetelmiä dimensioiden vähentämiseen on esimerkiksi vuonna 2008 esitelty t-distributed stochastic neighbor embedding (t-SNE) [4], jota käytetään erityisesti datan visualisointiin, sekä vuonna 2018 esitetty Uniform Manifold Approximation and Projection (UMAP) [5]. Näiden lisäksi uusia menetelmiä esitellään jatkuvasti. Eräs tällainen on UMAP:in kaltainen, vuonna 2020 julkaistu latent variable model (LVM) [6].

Datan määrä yhteiskunnassamme on lisääntynyt räjähdysmäisesti viimeisten vuosien aikana ja datan analysoinnista ja koneoppimisesta on tullut yhä tärkeämpää monissa sovelluskohteissa. Eri lähteistä saatavat data-aineistot ovat kuitenkin muodoltaan vaihtelevia ja sisältävät usein suuren määrän piirteitä toisistaan poikkeavilla arvoalueilla. Samanaikaisesti moni vasta-alkaja saa tehtäväkseen datan analysoinnin ja tekoälymenetelmien soveltamisen. Tämän kandintyön tavoitteena on vastata heidän ensimmäisen vaiheensa tarpeisiin.

Esikäsittelyn lisäksi työssä perehdytään datan analysointiin erityisesti visualisoinnin avulla. Visualisointi antaa tärkeää tietoa datan rakenteista ja sen avulla voidaan nähdä esimerkiksi datan jakautumista klustereihin.

Työssä esitellään erilaisia vaihtoehtoja datan skaalaamiseen ja normalisointiin sekä dimensioiden vähentämiseen. Teoriaa menetelmien taustalla avataan ja havainnollistetaan esimerkkien avulla, jotta menetelmien vaikutus oivalletaan. Esimerkkidatoina käytetään muun muassa säähavaintoja ja osakkeiden hintakehitystä. Menetelmien käsittelyn yhteydessä perehdytään niihin liittyviin ominaisuuksiin ja tuodaan lisäksi ilmi mahdollisia ongelmia. Lopussa esitellään prosessia, jolla uutta dataa on mahdollista käsitellä datan lataamisesta visualisointiin saakka.

Työssä on kehitetty Python-työkalu [7], jonka avulla datan käsittely ja visualisointi onnistuu graafisessa käyttöliittymässä. Ohjelmassa hyödynnetään scikit-learn Python-kirjaston tarjoamia funktioita muunnosten aikaansaamiseksi. Työkalun tavoitteena on johdattaa datan esikäsittelyyn ja visualisointiin erityisesti henkilöitä, joilla ei ole paljoa ohjelmointikokemusta.

2. SKAALAAMINEN JA NORMALISOINTI

Datan skaalaaminen tai normalisointi on tavallinen datan esikäsittelyn vaihe, jossa data muunnetaan niin, että piirteiden arvoalueet saadaan samalle lukuvälille tai noudattamaan tiettyä jakumaa. Koneoppimismallit näkevät datan vain numeroina ja tästä syystä suuret vaihtelut piirteiden arvoalueissa saattavat aiheuttaa jonkin piirteen korostumista, vaikka tämä ei erottelisikaan näytteitä yhtä hyvin kuin pienemmän arvoalueen piirteet. Piirteiden asettamisella samalle skaalalle saadaan siis yhdenmukaistettua niiden vaikutus laskennassa [8].

Arvoalueiden vaihtelu aiheuttaa haasteita erityisesti algoritmeissa, jotka määrittävät datapisteiden samankaltaisuutta etäisyysmittojen perusteella. Tällaisia etäisyyksiin perustuvia algoritmeja ovat esimerkiksi k-NN, K-Means ja SVM [9]. On tutkittu, että esimerkiksi K-Means -klusteroinnin tapauksessa luokittelutarkkuus voi parantua, kun data esikäsitellään sopivilla skaalauksilla tai normalisoinneilla [10].

Tarkastellaan esimerkkinä kuvan 1 tapausta, jossa datapistettä kuvaa kaksi piirrettä. Vasemmanpuoleisesta alkuperästä datasta huomataan nopeasti kuinka kahden piirteen arvoalueet poikkeavat toisistaan huomattavasti. Vasemmanpuoleisessa datassa nämä piirteet ovat normalisoitu, jolloin piirre x2 ei pääse vaikuttamaan datapisteiden etäisyyksiin suhteettoman suuresti.

	x1	x2
0	2	2200
1	4	2600
2	7	2600
3	12	3100

	x1	x2
0	-1.128330	-1.331543
1	-0.597351	-0.078326
2	0.199117	-0.078326
3	1.526564	1.488195

Kuva 1. Kuvassa vasemmalla puolella alkuperäinen ja oikealla puolella normalisoitu data.

Havainnollistetaan normalisoinnin vaikutusta vielä laskemalla pisteiden 1 ja 0, sekä 1 ja 2 väliset euklidiset etäisyydet kaavan (1) mukaisesti, jossa p_x ja q_x tarkoittavat tietyn datapisteen piirrettä x. Ennen normalisointia pisteiden 1 ja 0 välinen etäisyys oli 400 ja pisteiden 1 ja 2 välinen etäisyys 3. Normalisoinnin jälkeen taas vastaavat etäisyydet olivat 1.36 ja 0.80. Piirteiden normalisoinnin ansiosta pisteiden väliset etäisyydet ovat siis paljon paremmin verrattavissa.

$$\sqrt{(p_1 - q_1)^2 + (p_2 - q_2)^2} \tag{1}$$

Myös neuroverkkojen tapauksessa normalisoinnista on hyötyä, sillä verkkojen opetusprosessi voi tehostua huomattavasti piirteiden ollessa samoilla arvoalueilla [11].

Poikkeuksia syötedatan skaalauksen tai normalisoinnin tarpeeseen ovat vain satunnaismetsä- ja naivi Bayes -tekniikat, sillä ne käsittelevät kutakin piirrettä erikseen. Käytännössä kaikilla muilla menetelmillä on kuitenkin tarve jonkinlaiselle piirteiden normalisoinnille tai skaalaukselle.

Erilaisia skaalaus-ja normalisointimenetelmiä on olemassa lukuisia, eikä mikään niistä ole universaali, jokaiseen tilanteeseen sopiva ratkaisu. Sopivan menetelmän valitseminen onkin kiinni käyttäjästä ja vaatii tietoa sekä datasta, että koneoppimismenetelmästä, johon data syötetään. Yleisesti on hankala sanoa etukäteen mikä menetelmä tulee toimimaan parhaiten uuden datan tapauksessa ja sopivan menetelmän valinta vaatii vaihtoehtojen testausta.

Eri tutkimukset demonstroivat kuinka normalisointimenetelmän valinta voi johtaa huomattaviinkin eroihin luokittelutarkkuuksissa [12]-[15]. Epäsopivalla menetelmällä on mahdollista myös heikentää luokittelutarkkuutta [15]. Tämä osoittaa hyvin, kuinka menetelmän valinta on riippuvaista datasta. Seuraavaksi käsitellään neljää yleisesti käytettyä menetelmää. Yhtälöissä 2-5 $\widehat{\boldsymbol{x}}_{\boldsymbol{i}}$ on skaalattu/normalisoitu piirre ja $\boldsymbol{x}_{\boldsymbol{i}}$ alkuperäinen piirre.

2.1. Skaalaus lukuvälille

Skaalausmenetelmää, jossa piirteiden arvot skaalataan tietylle lukuvälille kutsutaan englanniksi yleisesti nimellä Min-Max scaling/normalization. Menetelmän tarkoituksena on muokata piirteitä niin, että niiden arvoalueet ovat samalla skaalalla. Min-Max skaalauksen etuna on se, että skaalaus säilyttää tarkasti kaikki alkuperäisessä datassa olevat näytteiden väliset suhteet [12]. Skaalaus halutulle välille $[\min x_{uusi}, \max x_{uusi}]$ saadaan yhtälön (2) mukaisesti.

$$\widehat{\boldsymbol{x}}_{\boldsymbol{i}} = \frac{\boldsymbol{x}_{\boldsymbol{i}} - \min(\boldsymbol{x}_{\boldsymbol{i}})}{\max(\boldsymbol{x}_{\boldsymbol{i}}) - \min(\boldsymbol{x}_{\boldsymbol{i}})} (\max x_{uusi} - \min x_{uusi}) + \min x_{uusi} \tag{2}$$

Yhtälön avulla saadaan siis skaalattua alkuperäinen arvoalueella $[\min(x_i), \max(x_i)]$ vaihteleva piirre x_i välille $[\min x_{uusi}, \max x_{uusi}]$. Piirteiden skaalaus välille [0,1] on yleisesti käyetty skaalauksen erikoistapaus ja saadaan yksinkertaisesti yhtälöllä (3).

$$\widehat{\boldsymbol{x}}_{\boldsymbol{i}} = \frac{\boldsymbol{x}_{\boldsymbol{i}} - \min(\boldsymbol{x}_{\boldsymbol{i}})}{\max(\boldsymbol{x}_{\boldsymbol{i}}) - \min(\boldsymbol{x}_{\boldsymbol{i}})} \tag{3}$$

Min-Max skaalauksen ongelmana on kuitenkin poikkeavien näytteiden aiheuttama varsinaisten luokkanäytteiden hajonnan pieneneminen.

2.2. Jakauman normalisointi

Jakauman normalisointi (engl. standardization) on datan normalisointimenetelmä, joka hyödyntää keskiarvoa ja keskihajontaa piirteiden normalisoimiseen. Keskiarvo ja keskihajonta lasketaan jokaisesta piirteestä erikseen. Muunnos lasketaan alla olevan yhtälön (4) mukaisesti.

$$\widehat{\boldsymbol{x}}_{\boldsymbol{i}} = \frac{\boldsymbol{x}_{\boldsymbol{i}} - \overline{\boldsymbol{x}}_{\boldsymbol{i}}}{\sigma_i}, \tag{4}$$

jossa $\overline{\boldsymbol{x}}_{\boldsymbol{i}}$ on piirteen $\boldsymbol{x}_{\boldsymbol{i}}$ keskiarvo. Muunnoksen tuloksena on datajoukko, jossa jokaisen piirteen keskiarvo on 0 ja varianssi 1. Jakauman normalisointi on tarpeen muun muassa

neuraalilaskennassa ja lisäksi hyödyllinen esimerkiksi logistisessa regressiossa ja tukivektorikoneissa.

2.3. Keskiarvon normalisointi

Keskiarvon normalisointi asettaa nimensä mukaisesti piirteen keskiarvon nollaksi. Menetelmässä jokaisesta piirteen arvosta vähennetään piirteen keskiarvo ja lopuksi jaetaan piirteen maksimi- ja minimiarvon erotuksella kuten yhtälössä (5).

$$\widehat{\boldsymbol{x}}_{\boldsymbol{i}} = \frac{\boldsymbol{x}_{\boldsymbol{i}} - \text{average}(\boldsymbol{x}_{\boldsymbol{i}})}{\max(\boldsymbol{x}_{\boldsymbol{i}}) - \min(\boldsymbol{x}_{\boldsymbol{i}})} \tag{5}$$

2.4. Yksikkövektorinormalisointi

Yksikkövektorinormalisointi poikkeaa kaikista edellä mainituista skaalaus- ja normalisointimenetelmistä, sillä se on rivioperaatio. Yksikkövektorinormalisoinnissa siis normalisoidaan piirteiden sijaan kukin piirrevektori yksikön mittaiseksi [16]. Piirrevektori sisältää yksittäisen näytteen piirteiden arvot. Yksikön mittainen vektori saadaan jakamalla piirrevektorin arvot piirrevektorin pituudella yhtälön (6) mukaisesti. Pituutena voidaan käyttää joko city-block etäisyyttä (L1 normi) tai euklidista etäisyyttä (L2 normi).

$$\widehat{\boldsymbol{x}} = \frac{\boldsymbol{x}}{\|\boldsymbol{x}\|}, \tag{6}$$

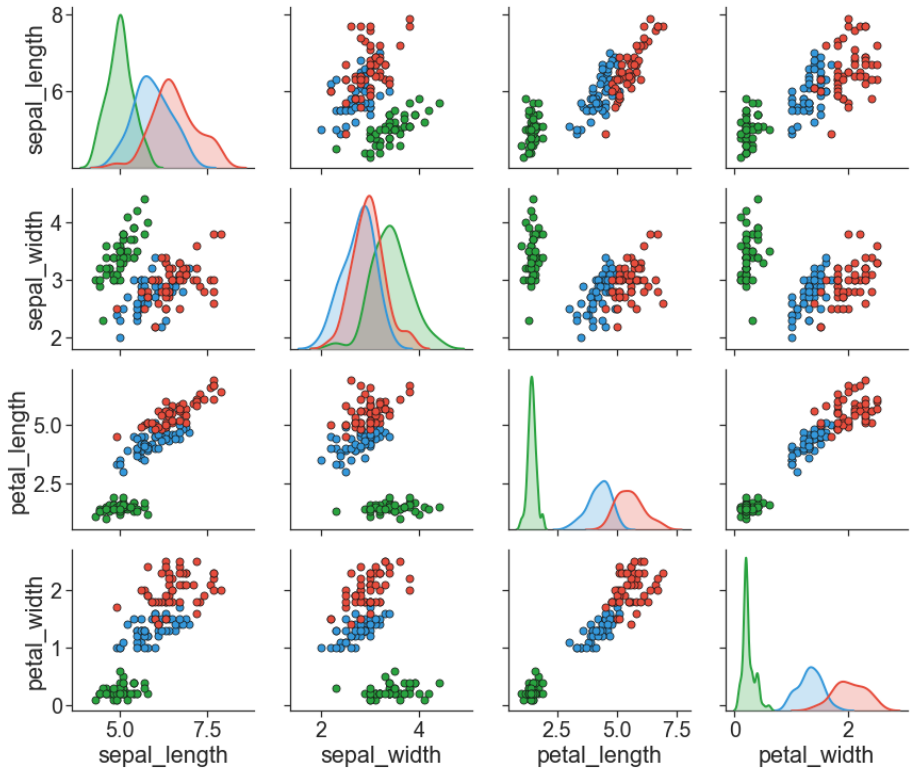
missä $\hat{x}$ on normalisoitu piirrevektori,
x on alkuperäinen piirrevektori ja
$\|x\|$ on piirrevektorin pituus

Yksikkövektorinormalisoinnin jälkeen piirrevektorin arvot ovat nollan ja yhden välillä.

3. DATAN VISUALISOINTI JA DIMENSIOIDEN VÄHENTÄMINEN

Datan visualisointi on yleisesti hyödynnetty tutkivan data-analyysin menetelmä, jonka avulla pyritään löytämään datan sisältämiä rakenteita. Tavallinen ongelma datan visualisoinnissa on kuitenkin datan sisältämien piirteiden suuri määrä. Ei ole harvinaista, että datassa on viisikymmentä tai satakin piirrettä ja esimerkiksi valokuvassa saattaa olla tuhansia pikseleitä, joka tarkoittaa tuhansia piirteitä. Dataa visualisoidessa voidaan esittää helposti kuitenkin vain kaksi tai kolme ulottuvuutta.

Kenties yksinkertaisin menetelmä korkeadimensioisen datan visualisoimiseen on generoida kaksiulotteisia pistekaavioita (engl. scatter plot), joissa visualisoidaan kahta piirrettä kerrallaan. Kuvat voidaan sitten koota pistekaaviomatriisiin [17], (Kuva 2) [18]. Havainnollisuuden lisäämiseksi datajoukon kolme luokkaa on eroteltu omilla väreillään. Matriisi on luotu Pythonin Seaborn-kirjaston avulla.



Kuva 2. Pistekaaviomatriisi iris-datajoukosta.

Matriisin diagonaalilla, jossa yksittäisen kuvan akselit olisivat samoja, voidaan esittää esimerkiksi tiheyskaavio (engl. density plot). Tiheyskaavio kuvaa datapisteiden jakaumaa ja antaa lisätietoa datan rakenteesta. Yksittäisissä pistekaavioissa datajoukon luokat on eroteltu toisistaan omilla väreillään havainnollisuuden lisäämiseksi. Mahdollisten ennalta tunnettujen luokkien erottaminen väreillä on yleisesti hyvä keino datan tulkinnan helpottamiseksi.

Vaikka pistekaaviomatriisi on joissain tapauksissa kohtalaisen hyvä keino datan visualisointiin, se ajautuu suuriin ongelmiin piirteiden määrän kasvaessa. Esimerkiksi jo kymmenellä piirteellä matriisin koko kasvaa niin suureksi, että siitä tulee hyvin hankalasti tulkittava.

Visualisoinnin lisäksi korkeadimensioinen data aiheuttaa ongelmia esimerkiksi etäisyysmittoihin perustuville luokittelijoille, sillä piirreavaruuden harveneminen heikentää etäisyysmittojen eroavaisuuksia. Tätä ilmiötä kutsutaan yleisesti nimellä dimensionaalisuuden kirous (engl. curse of dimensionality). Ensin piirteiden lisääminen parantaa luokittelijan tarkkuutta, mutta tietyssä pisteessä piirreavaruus alkaa harveta niin, että luokittelija ei enää onnistu erottelemaan dataa yhtä tarkasti.

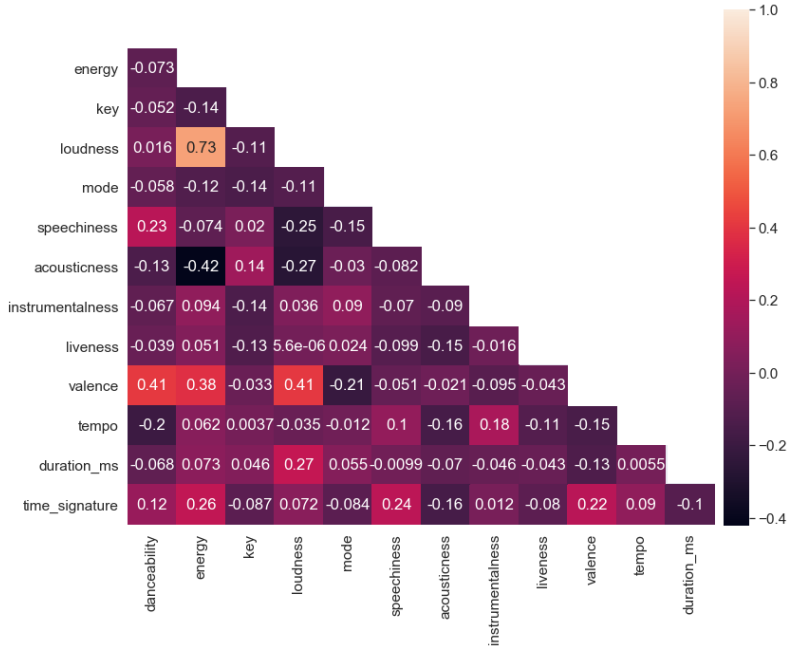
3.1. Dimensioiden vähentäminen

Kumpaakin edellä mainittua ongelmaa voidaan lähestyä dimensioiden vähentämisen avulla. Dimensioiden vähentäminen on keino, jossa korkeadimensioinen data kuvataan vähemmällä piirteiden määrällä niin, että mahdollisimman suuri osa alkuperäisen datan sisältämästä informaatiosta kyetään säilyttämään. Dimensioiden vähentäminen on siis kuitenkin aina häviöllinen operaatio. Dataa visualisoitaessa piirteiden lukumäärä pudotetaan kahteen tai kolmeen, jolloin datapisteet saadaan asetettua helposti ihmisen ymmärtämään pistekaavioon. Kaikkia dimensioiden vähentämismenetelmiä ei ole kuitenkaan alkujaan suunniteltu datan visualisointiin.

Toinen käyttötarkoitus dimensioiden vähentämiselle on heikosti dataa kuvaavien piirteiden poistaminen tai yhdistely siten, että mahdollisimman suuri osa datan sisältämästä informaatiosta saadaan kuvattua mahdollisimman pienellä piirteiden määrällä. Piirteiden vähentämisen motivaationa on laskennallisten kustannuksien pieneneminen ja esimerkiksi mahdollinen luokittelutarkkuuden paraneminen.

Dimensioiden vähentämistä voidaan lähestyä kahdella eri tavalla. Ensimmäinen vaihtoehto on hakea käytettäväksi parhaiten luokkia erottelevat piirteet (engl. feature selection). Toinen vaihtoehto on piirteiden irrotus (engl. feature projection, feature extraction), jossa tarkoituksena on tuottaa yhdistelmiä alkuperäisen datan muuttujista niin, että saadut piirteet kuvaavat edelleen dataa mahdollisimman tarkasti.

Piirteiden välisiä suhteita voi tutkia esimerkiksi korrelaatiomatriisin avulla (Kuva 3). Se kertoo yksinkertaisesti kuinka eri piirteet korreloivat toistensa kanssa.



Kuva 3. Korrelaatiomatriisi Spotifyn kuunnelluimpien kappaleiden ominaisuuksia sisältävästä datasta.

Korrelaatiomatriisi on yleisesti käytetty tutkivan data-analyysin menetelmä. Piirteiden mielekkyyden arvioinnissa korrelaatiomatriisi antaa tärkeää tietoa piirteiden korrelaatiosta sekä selitettävän muuttujan, että toistensa kanssa.

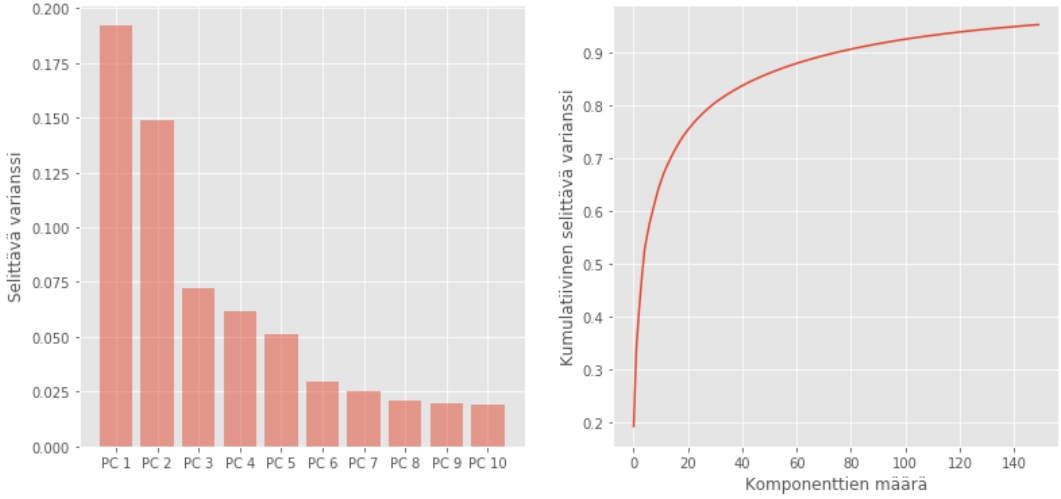
Vaikka oletettaisiin kaikkien piirteiden olevan relevantteja, piirteiden määrä voi silti olla suurempi kuin on tarpeen. Dimensioiden vähentämiseen on yleisesti käytettyjä muunnosalgoritmeja, jotka pakkaavat korkeadimensioista dataa niin, että alkuperäinen informaatio säilytetään mahdollisimman hyvin. Nämä menetelmät jakautuvat lineaarisiin ja epälineaarisiin [19]. Erilaisten menetelmien suuren kirjon vuoksi on hankala sanoa etukäteen, mikä niistä toimii parhaiten uuden datan tapauksessa. Menetelmät pyrkivät nimittäin säilyttämään datan eri ominaisuuksia ja saatu tulos riippuu paljolti käytetystä datasta. Usein parhaiten toimiva ratkaisu löydetäänkin vain kokeilemalla ja vertailemalla saatuja tuloksia.

Seuraavaksi tarkastellaan muutamaa dimensioiden vähennysmenetelmää. Tarkoituksena on valaista niiden keskinäisiä eroja ja samankaltaisuuksia, joiden tuntemisesta on hyötyä valittaessa ratkaisuja mahdollisiin sovelluksiin.

3.1.1. Pääkomponenttianalyysi

Pääkomponenttianalyysi (engl. principal component analysis, PCA) [1] on lineaarinen dimensionaalisuuden vähennysmenetelmä, jota käytetään usein ensimmäisenä menetelmänä uutta dataa visualisoitaessa. PCA:n tavoitteena on löytää korkeadimensioisesta datasta ortogonaaliset kantavektorit ns. pääkomponentit, joiden avulla alkuperäinen data voidaan esittää mahdollisimman vähäisellä informaation menetyksellä.

Pääkomponentit siis maksimoivat datan varianssin siten, että ensimmäinen pääkomponentti kuvaa mahdollisimman suuren osan informaation vaihtelusta ja siitä seuraavat komponentit kuvaavat jäljelle jäävää informaatiota vähenevässä järjestyksessä, ks. Kuva 4 (a). Tulokset on laskettu Labelled Faces in the Wild-datajoukosta [20], jossa jokaista datapistettä kuvasi 1850 piirrettä. Kuvassa 4 (b) nähdään, kuinka tämän datajoukon varianssista on saatu säilytettyä yli 90 prosenttia vain 80 ensimmäisen pääkomponentin avulla.



(a) (b)

Kuva 4. Varianssin jakautuminen ensimmäisten pääkomponenttien välillä.

PCA on suosittu myös esikäsittelymenetelmänä korkeadimensioiselle datalle. Tutkimukset ovat osoittaneet, että PCA:n käyttö esikäsittelyssä on parantanut luokittelutuloksia datan ollessa korkeadimensioista [21]-[23].

Matemaattisesti pääkomponentit saadaan laskemalla datajoukon $\boldsymbol{X} \in \mathbb{R}^{d \times N}$ kovarianssimatriisin $\boldsymbol{C_x}$ ominaisarvot λ, sekä ominaisvektorit $\boldsymbol{e}$ [24].

$$\boldsymbol{C_x e} = \boldsymbol{\lambda e} \tag{7}$$

Datapisteiden $\boldsymbol{x_i}$ dimensio saadaan vähennettyä mataladimensioiseksi esitykseksi $\boldsymbol{y_i}$ laskemalla pistetulo matriisin $\boldsymbol{W}$ kanssa. Matriisi $\boldsymbol{W}$ sisältää suurimpia ominaisarvoja vastaavat ominaisvektorit.

$$\boldsymbol{y_i} = \boldsymbol{W x_i} \tag{8}$$

Pääkomponenttianalyysin tulokset ovat helposti visuaalisesti tulkittavissa ja menetelmä toimii hyvin, kun datan varianssi keskittyy muutamalle eri suunnalle. Muiden lineaaristen menetelmien tapaan se kuitenkin vaatii datan riippuvuuksien lineaarista selitettävyyttä. Paljon epälineaarisia riippuvuuksia sisältävän datan tapauksessa muunnokset eivät onnistu kovin hyvin.

Kuvassa 5 on laskettu kaksiulotteisesta datasta pääkomponentit 1 ja 2, jonka jälkeen data on kuvattu näiden pääkomponenttien mukaisilla akseleilla. Lisäksi kuvassa 5 (c) on esitetty data pelkästään ensimmäisen pääkomponentin avulla yhdessä ulottuvuudessa.

(a) (b) (c)

Kuva 5. Pääkomponenttianalyysi kaksidimensioisen datan tapauksessa.

3.1.2. Monidimensioskaalaus

Monidimensioskaalauksella (engl. Multidimensional scaling, MDS) viitataan menetelmiin, jotka analysoivat datapisteiden välisiä samankaltaisuuksia ja pyrkivät säilyttämään nämä mahdollisimman hyvin. Ensimmäinen askel monidimensioskaalauksessa on etäisyysmatriisin muodostaminen. Etäisyysmatriisi

voidaan laskea esimerkiksi pisteiden välisien euklidisten etäisyyksien avulla, mutta myös muita menetelmiä on olemassa.

Jos etäisyysmatriisi lasketaan euklidisten etäisyyksien avulla, puhutaan tällöin klassisesta skaalauksesta, joka on lineaarinen menetelmä [2]. Lineaarinen MDS on hyvin lähellä pääkomponenttianalyysiä ja on osoitettu, että samaan aliavaruuteen skaalattaessa, näiden kahden menetelmän tuottamat tulokset vastaavat toisiaan [25]. Lineaarisessa monidimensioskaalauksessa skaalatut pisteet saadaan etäisyysmatriisin ominaisarvojen- ja vektorien avulla.

Epälineaaristen riippuvuuksien löytäminen datasta vaatii kuitenkin erilaista etäisyysfunktiota. Yleisesti epälineaarisiin MDS menetelmiin liittyy käsite häviöfunktio (engl. cost function). Häviöfunktion tarkoituksena on parantaa sovituksen onnistumista minimoimalla dimensioiden välisten etäisyyksien neliöityjen virhevektorien summa. Eräs perinteinen epälineaarinen monidimensioskaalausmenetelmä on mitallinen MDS (engl. metric MDS). Mitallinen MDS on yksinkertaisin monidimensioskaalausmenetelmä ja sen häviöfunktio saadaan kaavan (9) mukaisesti [26].

$$E = \sum_{i<j} (\boldsymbol{d_{ij}} - \widehat{\boldsymbol{d}}_{\boldsymbol{ij}})^2 \tag{9}$$

jossa $\boldsymbol{d_{ij}}$ tarkoittaa datapisteiden etäisyyttä alkuperäisessä ulottuvuudessa ja $\widehat{\boldsymbol{d}}_{\boldsymbol{ij}}$ etäisyyttä saadussa ulottuvuudessa.

Muita variaatioita monidimensioskaalauksesta on esimerkiksi epämetrinen MDS (engl. Non metric MDS) ja Sammonin kuvaus [27]. Kaikki nämä menetelmät noudattavat kuitenkin samaa peruskaavaa häviöfunktioidensa suhteen. Sammonin kuvauksessa lyhyemmät etäisyydet saavat suuremman painoarvon yhtälön (10) häviöfunktiosta johtuen.

$$E = \frac{1}{\sum_{\boldsymbol{i<j}} \boldsymbol{d_{ij}}} \sum_{\boldsymbol{i<j}} \frac{(\boldsymbol{d_{ij}} - \hat{\boldsymbol{d}}_{\boldsymbol{ij}})^2}{\boldsymbol{d_{ij}}} \tag{10}$$

Epämetrinen MDS taas muokkaa pisteiden välisiä etäisyyksiä funktiolla, joka kuvaa pisteiden etäisyyksien välistä järjestystä niiden varsinaisen etäisyyden sijaan.

3.1.3. t-SNE

T-distributed Stochastic Neighbor Embedding (t-SNE) [4] on vuonna 2008 kehitetty epälineaarinen dimensioiden vähennysmenetelmä, jota käytetään erityisesti datan visualisointiin kahdessa tai kolmessa ulottuvuudessa. T-SNE kuvaa korkeadimensioiset datapisteet matalampaan ulottuvuuteen siten, että lähempänä olevat datapisteet sijaitsevat lähellä toisiaan korkealla todennäköisyydellä myös matalammassa ulottuvuudessa ja kauempana olevat taas kaukana. Lisäksi t-SNE pyrkii säilyttämään datassa olevat klusterit siten, että ne erottuvat selkeästi toisistaan. Klusteroitumiseen vaikuttaa kuitenkin suuresti käyttäjän algoritmille antamat parametrit. T-SNE on saavuttanut paljon suosiota viime aikoina ja se on ollut käytössä esimerkiksi tietoturva- [28] ja syöpätutkimuksessa [29].

T-SNE algoritmin toiminta koostuu kahdesta osasta. Ensin t-SNE muodostaa datapisteiden välisen todennäköisyysjakauman, jossa saadut arvot määräytyvät pisteiden välisen samankaltaisuuden perusteella. Samankaltaisuutta mitataan alkuperäisessä algoritmissa euklidisen etäisyyden avulla, mutta myös muita etäisyysmittoja on mahdollista käyttää. Todennäköisyys on sitä suurempi mitä samankaltaisempia datanäytteet ovat. Tämän jälkeen ne sijoitetaan alempaan ulottuvuuteen samankaltaisuuden todennäköisyysjakauman avulla jakaumien välinen Kullback-Leibler divergenssi minimoiden.

T-SNE siis minimoi divergenssin kahden todennäköisyysjakauman välillä: jakauman, joka mittaa alkuperäisten datapisteiden parittaista samankaltaisuutta ja jakauman, joka mittaa muunnoksessa saatujen datapisteiden parittaista samankaltaisuutta [30]. Näin sekä korkeamman että matalamman ulottuvuuden datapisteiden todennäköisyysjakaumat vastaavat mahdollisimman hyvin toisiaan.

3.1.4. Isomap

Isomap (isometric feature mapping) on epälineaarinen dimensioiden vähentämismenetelmä, joka ottaa huomioon datapisteiden välisen suoran euklidisen etäisyyden sijaan etäisyyden monistolla (engl. manifold) [3]. Isomap siis olettaa datan olevan jakautunut alempidimensioiselle monistolle.

Ensimmäinen vaihe muunnoksen laskemisessa on määrittää naapuripisteet monistolla M. Naapuripisteet määritetään pisteiden välisen geodeettisen etäisyyden avulla. Geodeettiset etäisyydet lasketaan naapurigraafin avulla, jossa jokainen piste liitetään k-lähimpään naapuriinsa. Graafista saadut etäisyydet ovat siis approksimaatioita todellisista geodeettisista etäisyyksistä. Graafin perusteella Isomap arvioi pisteiden väliset etäisyydet etsimällä lyhimmän polun pisteiden välillä graafissa ja sijoittamalla näin saadut geodeettiset etäisyydet etäisyysmatriisiin. Viimeisessä vaiheessa datapisteet sijoitetaan d-dimensioiseen euklidiseen avaruuteen klassisen monidimensioskaalauksen avulla, jolloin alkuperäisen moniston geometria säilytetään mahdollisimman hyvin. Isomap eroaa klassisesta monidimensioskaalauksesta siis vain sen erilaisella tavalla määrittää pisteiden väliset etäisyydet.

3.1.5. UMAP

Vuonna 2018 julkaistu Uniform Manifold Approximation and Projection (UMAP) [5] on toinen viime aikoina paljon huomiota saanut epälineaarinen dimensioiden vähentämismenetelmä t-SNE:n lisäksi. Se pyrkii säilyttämään korkeadimensioisessa datassa olevat topologiset ominaisuudet ja sitä voidaan käyttää datan visualisoinnin lisäksi myös yleiseen epälineaariseen dimensioiden vähentämiseen. UMAP-algoritmin etuna on hyvän visualisointikyvyn lisäksi sen nopea laskenta-aika suuremmillakin datajoukoilla [31].

Algoritmi toimii siten, että korkeadimensioisen datan näytteille muodostetaan sumea topologinen esitystapa. Jokaisen datapisteen ympärille määritetään k-lähimmän naapurin sumea simpleksi (engl. fuzzy simplicial complex). Käytännössä UMAP muodostaa siis k-lähimmän naapurin välisen painotetun graafin, joka kuvaa todennäköisyyttä, jolla pisteet ovat yhdistetty. Graafin muodostumiseen vaikuttaa myös säde, jonka sisällä naapureita tutkitaan. UMAP tekee painotetusta graafista sumean vähentämällä pisteiden välisen yhteyden todennäköisyyttä säteen kasvaessa.

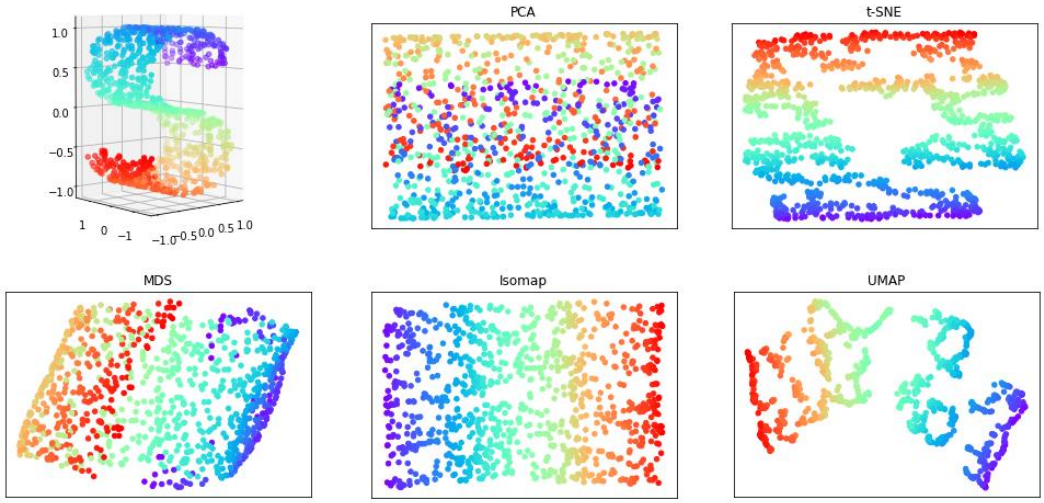
Säteen valinnan kanssa on oltava tarkkana, sillä liian suuri säteen arvo voi liittää jopa kaikki pisteet yhteen graafissa. Graafin muodostamisen jälkeen algoritmi pyrkii löytämään graafista alempidimensioisen esityksen mahdollisimman tarkasti minimoimalla korkean ja matalan dimension esitystapojen välisen ristientropian.

3.2. Menetelmien arviointi

Dimensioiden vähentämismenetelmiä on hyvin paljon ja ne antavat erilaisia tuloksia syötetystä datasta riippuen. Tämän takia uuden datan tapauksessa onkin usein tarpeen kokeilla monia erilaisia menetelmiä ja analysoida saatuja tuloksia. Etenkin visualisoinnin näkökulmasta saatujen lopputulosten vertaileminen on kuitenkin hankalaa, sillä visualisoinnin onnistumista on vaikea mitata. Saatuja tuloksia voi kuitenkin havainnoida empiirisesti ja tutkia, kuinka esimerkiksi mahdolliset klusterit erottuvat toisistaan.

Tämän lisäksi dimensioiden vähentämisen onnistumista voidaan mitata opettamalla luokittelija alkuperäiselle sekä dimensioltaan vähennetyille datoille ja vertailla saatuja tuloksia. Tämä sopii erityisesti silloin, kun haetaan sopivaa dimensioiden lukumäärää. Edellisten lisäksi paljon muitakin menetelmiä visualisointien arvioimiseksi on esitetty [32].

Kuvassa 6. voidaan nähdä, kuinka eri dimensioiden vähentämismenetelmät ovat toimineet, kun kolmidimensioinen s-muotoinen data on pudotettu kahteen dimensioon. Alkuperäisessä datassa on 1000 datapistettä, jotka ovat jakautuneet kaksiulotteiselle s-muotoiselle monistolle. Kuvasta huomataan esimerkiksi, kuinka Isomap on onnistunut säilyttämään alkuperäisen datan rakenteen geodeettisten etäisyyksien avulla ja kuvaa tarkasti datapisteiden etäisyydet monistolla.



Kuva 6. Kolmidimensioisen datan visualisointi kahdessa ulottuvuudessa eri menetelmillä suoritetun dimensioiden pudotuksen jälkeen.

PCA:n tapauksessa huomataan, kuinka lineaarisella menetelmällä on tyypillisesti vaikeuksia erotella etäisyyksiä datan ollessa jakautunut epälineaariselle monistolle.

Visualisoinnissa PCA on kuvannut punaiset ja violetit pisteet lähelle toisiaan, vaikka alkuperäisessä datassa ne sijaitsevat moniston vastakkaisissa päissä.

UMAP ja t-SNE taas keskittyvät enemmän datan lokaaleihin rakenteisiin ja pyrkivät klusteroimaan dataa, mutta säilyttävät kuitenkin yleisen datan järjestyksen monistolla. Etenkin UMAP:in ja t-SNE:n tapauksessa visualisoinnin lopputulokseen vaikuttaa huomattavasti myös algoritmeille annetut parametrit, kuten k-lähimmän naapurin valinta.

Vaikka edellisessä esimerkissä jotkin menetelmät toimivat toisia paremmin alkuperäisen moniston kuvaamisessa kaksiulotteiseksi, tämä ei tarkoita, että ne toimivat parhaiten myös uusien datajoukkojen tapauksessa. Pääkomponenttianalyysistä on hyvä myös painottaa, että sitä ei ole varsinaisesti suunniteltu datan visualisointia varten. PCA on kuitenkin tavallinen esikäsittelyn vaihe myös datan visualisoinnissa. Esimerkiksi ennen visualisointia t-SNE:n avulla, alkuperäisen datan piirteitä voidaan vähentää jo PCA:lla lopputuloksen parantamiseksi.

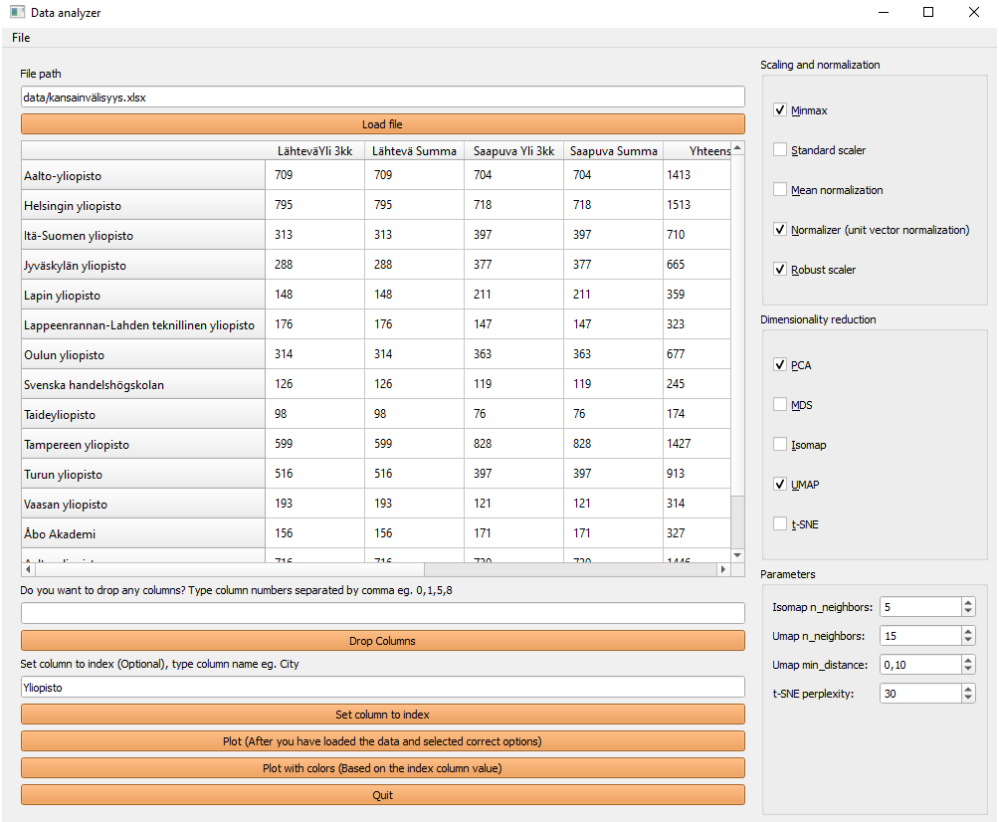
4. TODELLISEN DATAN KÄSITTELY JA VISUALISOINTI

Ennen koneoppimismenetelmien hyödyntämistä on tärkeää, että käyttäjällä on tietoa datan rakenteesta. Usein alkuperäinen data on hyvin moninaista ja sisältää arvoja vaihtelevilla skaaloilla. Uuden datan tapauksessa joudutaankin usein aluksi esikäsittelemään dataa ja visualisoimaan sitä sopivilla menetelmillä.

Edellisissä kappaleissa esiteltiin erilaisia menetelmiä datan skaalaukseen ja normalisointiin, sekä visualisointiin dimensioita vähentämällä. Tässä kappaleessa yhdistetään nämä datan käsittelyn ja tutkimisen vaiheet todellisille datajoukoille ja havaitaan kuinka eri menetelmät tuottavat erilaisia lopputuloksia. Samalla esitellään tutkielman kirjoitusprosessin aikana opetustarkoituksiin kehitetty Python-kielinen työkalu, jonka avulla dataa voidaan visualisoida käyttöliittymässä.

4.1. Visualisointityökalu

Datan käsittelyä ja visualisointia varten kehitetyn ohjelmistotyökalun tarkoituksena on helpottaa prosessia, jossa käyttäjä haluaa tutustua uuden datajoukon rakenteeseen. Ohjelma on toteutettu Python-ohjelmointikielellä ja se hyödyntää scikit-learn kirjaston tarjoamia funktioita skaalauksiin, normalisointeihin, sekä dimensioiden vähentämiseen. Lisäksi Pythonin pandas-kirjaston avulla työkalussa voidaan tarkastella dataa taulukkomuodossa ja muokata sitä. Ohjelman käyttöliittymä on kuvassa 7.

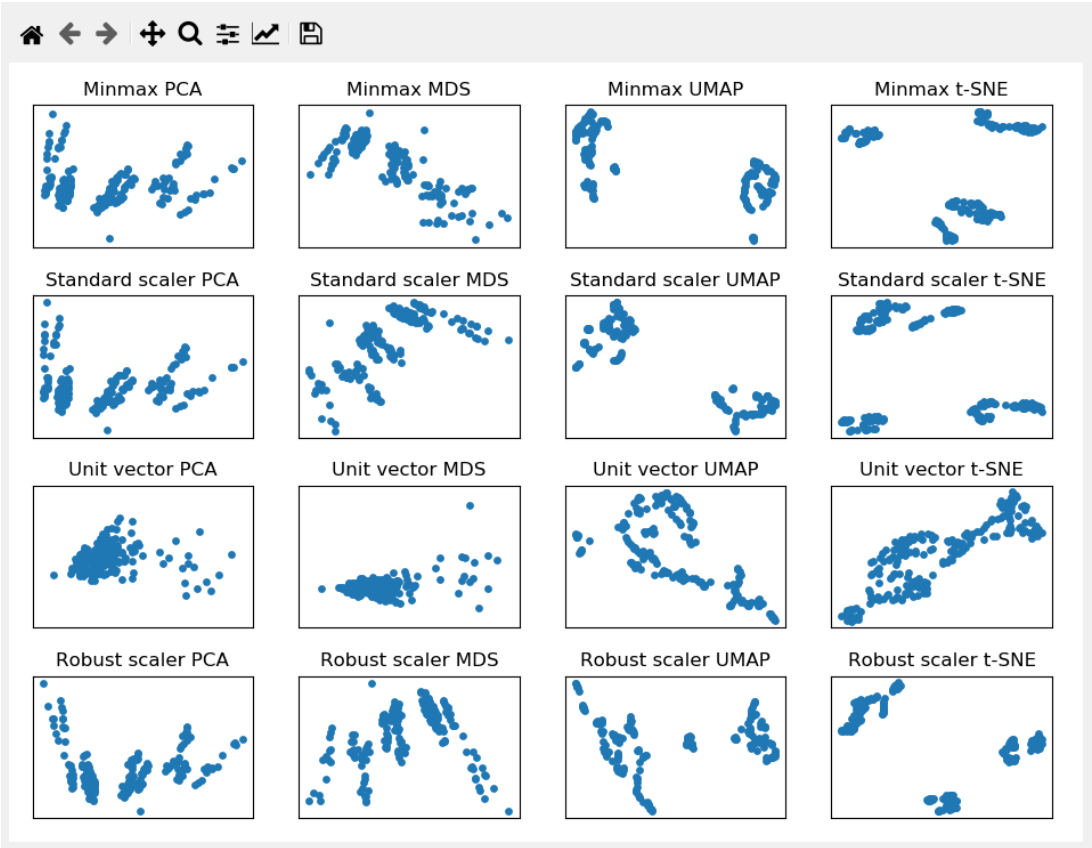


Kuva 7. Visualisointityökalun aloitusnäkymä, kun tiedosto on ladattu.

Ohjelman käyttö aloitetaan tuomalla tiedosto ohjelmalle. Ohjelma tukee excel-tiedostoja ja muutamia muita yleisiä tiedostomuotoja kuten .csv, .txt ja .data. Tiedoston lataamisen jälkeen data näkyy käyttäjälle suoraan taulukkomuotoisena ja käyttäjä voi halutessaan poistaa datajoukosta sarakkeita niiden indeksien perusteella. Lisäksi käyttäjä voi asettaa haluamansa indeksisarakkeen kuvaamaan näytteitä.

Ennen visualisointia on valittava halutut skaalaus, normalisointi ja dimensioiden vähentämismenetelmät. Vaihtoehtoina ovat skaalauksista ja normalisoinneista minmax-skaalaus, jakauman normalisointi, keskiarvon normalisoniti, yksikkövektorinormalisointi sekä scikit-learnin Robust Scaler, joka pyrkii lisäksi vähentämään poikkeavien näytteiden merkitystä.

Kuvassa 8 on esitetty esimerkki ohjelman tuottamasta visualisoinnista. Syötteenä on käytetty opetushallinnon tilastopalvelusta koostettua dataa Suomen korkeakoulujen kansainvälisyystilastoista [33].



Kuva 8. Visualisointityökalun tuottama kuva opiskelijadatalle.

Dimensioiden vähentämiseen on valittavana PCA, MDS, Isomap, UMAP ja t-SNE. Ohjelma tarjoaa myös muutamia yleisimpiä vaihtoehtoja dimensioiden vähentämismenetelmien parametrien muuttamiseen. Käyttäjä voi valita esimerkiksi, kuinka montaa lähinaapuria hän haluaa Isomap-, UMAP- ja t-SNE- algoritmien tarkastelevan.

Lopuksi kun kaikki tarpeelliset valinnat on tehty, käyttäjä voi valita haluaako visualisoida dataa siten, että eri luokkien näytteet on eroteltu toisistaan omilla väreillään vai ilman värejä. Visualisointi ilman värejä sopii esimerkiksi tapauksiin, jossa luokkatietoa ei ole saatavilla. Datan visualisoiminen värien avulla vaatiikin sen,

että käyttäjä on ensin asettanut datajoukolle indeksisarakkeen, sillä värit määräytyvät indeksisarakkeen arvon mukaan.

Plot-toiminnot suorittavat automaattisesti myös datan käsittelyä ennen visualisointia. Ne poistavat esimerkiksi datasarakkeista ylimääräiset numeroiden seassa olevat merkit ja muuttavat datatyypin numeeriseksi. Nykyisellään työkalu ei tunnista kuitenkaan luokitettuja muuttujia datasta ja se soveltuu vain jatkuva-arvoisten muuttujien käsittelyyn. Jos luokitettuja muuttujia kuitenkin esiintyy datassa, käyttäjän on arvioitava niiden käsittelyä ohjelman ulkopuolella tai jätettävä ne pois laskennasta ennen visualisointia.

4.2. Kokonaisuuden käsittely

Uuden datan käsittely on hyvä aloittaa yleensä tarkastelemalla raakadataa taulukkomuodossa. Taulukosta huomaa nopeasti millaisista piirteistä data koostuu ja miten esimerkiksi niiden arvoalueet vaihtelevat. Pythonissa tämä onnistuu esimerkiksi pandas-kirjastolla, jonka avulla data voidaan kuvata niin sanottuun dataframe muotoon, joka on verrattavissa esimerkiksi excel-taulukkoon.

Pandas-kirjasto tarjoaa paljon hyödyllisiä funktioita, joiden avulla datasta saa myös muuta tietoa, kuten puuttuvien arvojen määrän sarakkeissa, sekä sarakkeen datatyypin. Puuttuvien arvojen käsittely on tärkeää ennen algoritmien käyttöä, sillä se voi vaikuttaa saatuihin tuloksiin [34]. Kaikki algoritmit eivät osaa myöskään käsitellä puuttuvia arvoja. Lisäksi sarakkeiden on oltava numeerisessa muodossa, jotta niillä voidaan laskea.

Kun data on laskettavassa muodossa, se voidaan normalisoida ja pudottaa kahteen tai kolmeen dimensioon visualisointia varten. Normalisointia ja etenkin dimensioiden vähentämistä on syytä kokeilla erilaisilla menetelmillä.

Kuvasta 8 huomataan, kuinka eri menetelmät tuottavat myös tässä tapauksessa hyvin erilaisia visualisointeja. Alkuperäisessä datajoukossa kukin datapiste kuvaa tietyn vuoden tilastotietoja, joka saa PCA:n ja MDS:n tapauksessa datapisteitä asettumaan jonoiksi vuosittaisten arvojen hieman muuttuessa. UMAP ja t-SNE taas kokoavat dataa erinäisiksi klustereiksi skaalauksesta ja normalisoinnista riippuen.

Seuraavaksi tarkastellaan vielä kahta esimerkkiä reaalimaailman datan käsittelystä eri skaalaus- ja normalisointimenetelmien vaikutuksen havainnollistamiseksi. Jos tuloskuvissa on ryppäitä, niin esikäsittely todennäköisesti tukee datan rakenteiden koneoppimista. Lopullinen varmuus kuitenkin saavutetaan vasta esikäsittelyn ja koneoppimismenetelmän yhteiskäytöllä.

4.2.1. Säädata

Säähavaintojen analysointi on eräs mielenkiintoinen koneoppimisen sovelluskohde. Tarkastellaan tässä esimerkissä Ilmatieteenlaitoksen avoimen datan palvelusta [35] ladattua säädataa Oulunsalon mittauspisteeltä. Datajoukko sisältää päivittäisiä säähavaintoja vuoden 2010 alusta tiedoston latauspäivään 4.8.2020. Alkuperäisen data-aineiston sisältämät piirteet voi nähdä taulukossa 1.

Taulukosta huomataan selkeästi, että kaikki datassa olevat piirteet eivät ole visualisoinnin kannalta mielekkäitä. Käytämmekin datan visualisointiin vain viittä viimeisintä piirrettä: sademäärää, lumen syvyyttä, ilman lämpötilaa, sekä alinta- ja

ylintä lämpötilaa. Datan esikäsittelynä poistettiin lisäksi muutamia puuttuvia arvoja sisältäneitä näytteitä.

Taulukko 1. Ote Oulunsalon mittauspisteen säädata-aineistosta.

	Vuosi	Kk	Pv	Klo	Aikavyöhyke	Sademäärä (mm)	Lumensyvyys (cm)	Ilman lämpötila (degC)	Ylin lämpötila (degC)	Alin lämpötila (degC)
0	2010	1	1	00:00	UTC	0.4	22.0	-12.6	-9.1	-15.1
1	2010	1	2	00:00	UTC	1.0	22.0	-21.5	-14.6	-28.3
2	2010	1	3	00:00	UTC	0.6	24.0	-15.6	-13.8	-25.3
3	2010	1	4	00:00	UTC	0.4	29.0	-18.0	-14.0	-20.9
4	2010	1	5	00:00	UTC	1.0	27.0	-18.8	-15.9	-22.3
5	2010	1	6	00:00	UTC	-1.0	27.0	-21.9	-15.3	-29.6
6	2010	1	7	00:00	UTC	-1.0	26.0	-31.0	-29.0	-32.3
7	2010	1	8	00:00	UTC	3.7	25.0	-28.9	-23.9	-33.5
8	2010	1	9	00:00	UTC	-1.0	25.0	-21.5	-17.0	-26.7
9	2010	1	10	00:00	UTC	1.4	24.0	-5.6	-2.8	-17.6

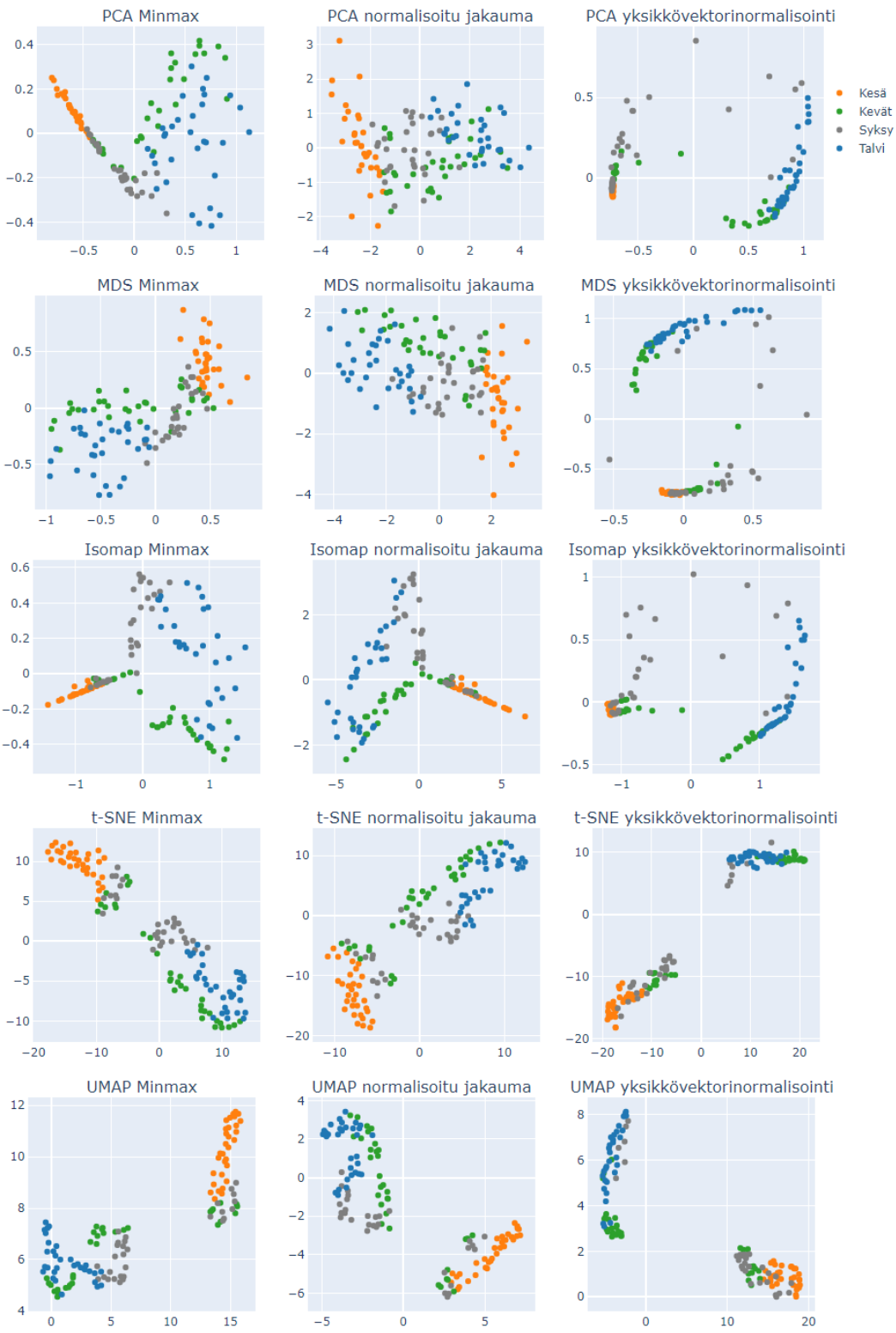
Lisäksi datapisteiden määrän karsimiseksi ja erottelun parantamiseksi aineiston perusteella laskettiin uusi datajoukko, jossa jokainen datapiste kuvaa alkuperäisistä piirteistä laskettuja kuukauden keskiarvoja. Kuvassa 9 nähdään kuinka eri menetelmät onnistuvat löytämään rakenteita datasta. Datapisteet on värjätty näytteen vuodenajan mukaisesti, jotta kuvista huomattaisiin helpommin mitkä datapisteet erottuvat toisistaan.

Eri menetelmien tuloksia tarkastelemalla huomataan, miten vuodenaikojen näytteet erottuvat toisistaan ja erityisesti t-SNE ja UMAP klusteroivat skaalattua ja normalisoitua dataa. Data näyttää jakautuvan erityisesti kesä- ja talvikuukausien välillä, johon todennäköisesti vaikuttaa erityisesti lumen syvyyden poikkeavuus nollasta. Myös PCA:n ja MDS:n tuloksista huomataan kuinka eri vuodenajat erottuvat toisistaan.

Myös erilainen skaalaus tai normalisointi muuttaa visualisoinnin lopputulosta, vaikka dimensioiden vähennysmenetelmä pysyykin samana. Suurin eroavaisuus nähdään yksikkövektorinormalisoinnin tuloksissa, joissa data klusteroituu selkeimmin. Myös PCA ja MDS jakavat yksikkövektorinormalisoituja näytteitä selkeämmin klustereihin, kun taas Minmax-skaalatulla ja normalisoidun jakauman datalla ne eivät erottuneet.

Yksikkövektorinormalisointi tuottaa rivioperaationa edellisistä menetelmistä huomattavasti poikkeavan piirteiden sekä piirrevektorien arvojakauman ja kahdessa dimensiossa datapisteet näyttävät asettuvan jonoiksi. PCA:n ja MDS:n tuloksista karkeasti hahmotettava klustereiden muodostama ympyrän muoto näyttäisi kuvaavan vuodenaikojen vaihtelua. Talvi- ja kesäkuukausien välisen erottelun lisäksi talvikuukausien näytteet hajaantuvat yksikkövektorinormalisoinnin tuloksissa selkeimmin, joka johtuu todennäköisesti lumensyvyyden vaihtelun aiheuttamasta erottelun parantumisesta.

Minmax ja erityisesti jakauman normalisointi erottelevat yksittäisiä näytteitä enemmän erilleen toisistaan ja näytteiden väliset etäisyydet vaihtelevat eri tavalla yksikkövektorinormalisointiin verrattuna. Kahdessa dimensiossa pistejakaumat asettuvatkin laajemmalle pinnalle ja myös värjätyt neljä eri luokkaa erottuvat kuvissa paremmin toisistaan menetelmien erilaisen luonteen takia.



Kuva 9. Säädatan visualisointi eri menetelmillä.

4.2.2. Osakedata

Koneoppimisen hyödyntäminen on tavallista myös osakemarkkina-analyysissä ja esimerkiksi osakkeiden hintakehityksen ennustamisessa. Osakedatan perusteella on mahdollista rakentaa esimerkiksi suosittelijoita tiettyjen osakkeiden ostamiselle ja myymiselle [36]. Tässä esimerkissä tutustutaan kuitenkin pelkkään datan visualisointiin.

Osakedataa on mahdollista visualisoida monella eri tavalla. On mahdollista keskittyä pelkkään hintakehitykseen ja tarkastella hintakehitystä ajan funktiona. Osakkeita voidaan myös klusteroida niiden tunnuslukujen perusteella. Tässä esimerkissä lähestytään ongelmaa kuitenkin hieman eri tavalla ja tutkitaan kuinka osakkeet asettuvat kuvaajaan, kun piirteinä on aikasarjaa viikoittaisesta hintakehityksestä. Alkuperäisen datan lähteenä on UCI Machine Learning Repository, johon datan on ladannut tohtori Michael Brown [36]. Datajoukko sisältää Dow Jones indeksin osakkeiden hintatietoja kahden kvartaalin ajalta. Datan piirteet siis kuvaavat osakkeiden viikoittaisesta käyttäytymisestä, kuten avaus- ja päätöshinnat.

Viikon avaus- ja päätöshintojen perusteella laskettiin jokaisen tarkastellun viikon osakekohtainen hinnan muutos, joista muodostettiin uusi taulukon 2 mukainen datajoukko. Jokainen sarake siis kuvaa yhden viikon hinnan muutosta. Yhteensä datajoukossa on 30 näytettä, joista jokaista kuvaa 25 piirrettä.

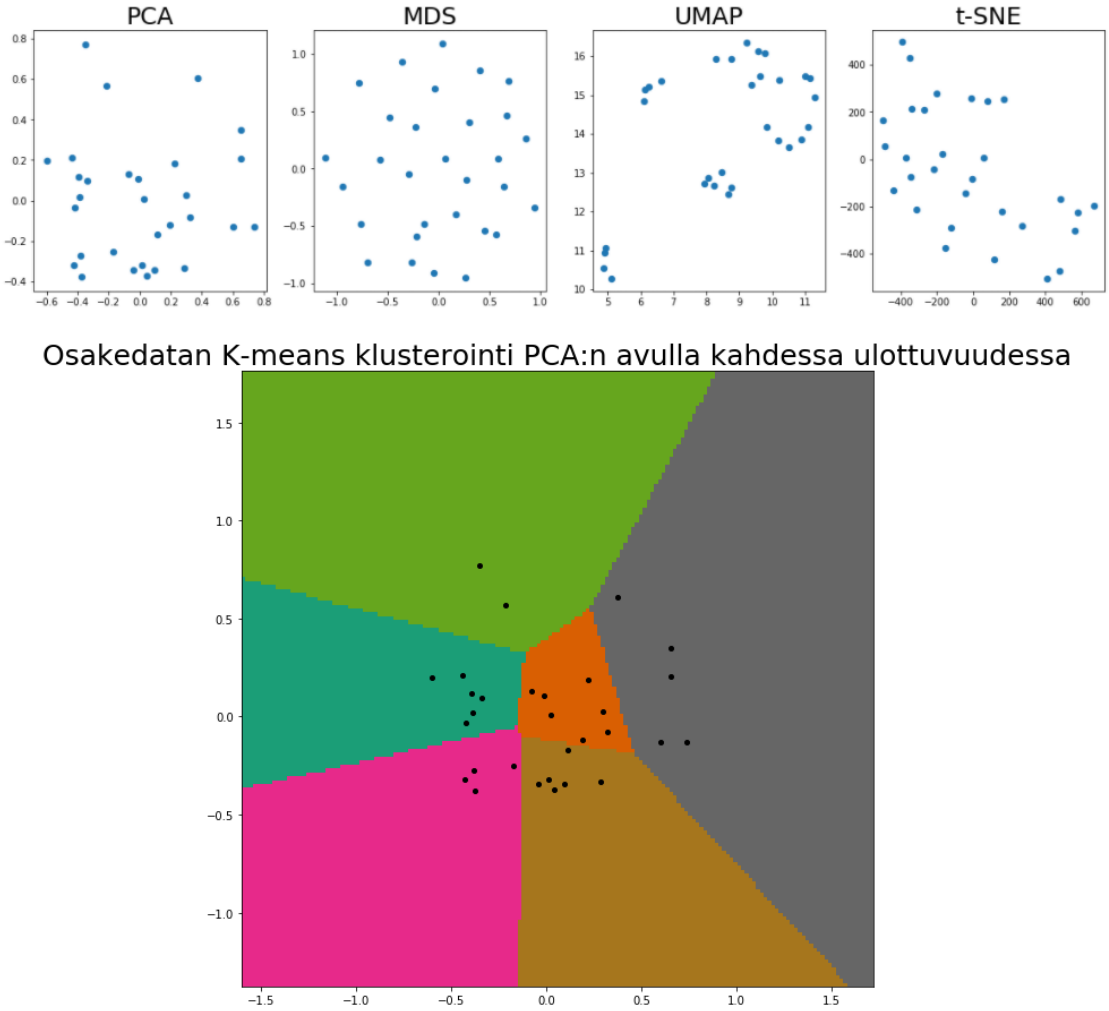
Taulukko 2. Ote osakedatajoukon näytteistä.

	0	1	2	3	4	5	6	7	8	9	...	15	16	17	18	19	20	21	22	23	24
AA	0.60	-0.74	-0.40	0.26	0.96	0.04	-0.11	-0.30	-0.23	-0.55	...	0.61	0.06	-0.12	-0.06	-0.74	0.52	-0.81	-0.64	-0.57	0.56
AXP	1.06	2.05	-0.03	-2.19	-0.31	2.79	-0.89	-1.41	-0.01	0.42	...	1.26	1.89	0.83	-0.57	1.77	0.39	-2.11	-1.39	0.54	-0.01
BA	3.23	0.65	0.82	-2.29	2.12	0.71	0.34	-0.05	-0.67	0.04	...	2.72	4.27	-1.04	-0.28	-1.14	0.44	-2.96	-2.07	1.10	-2.70
BAC	0.40	1.08	-0.83	-0.65	0.58	0.26	-0.02	-0.18	-0.15	0.20	...	-0.28	-0.04	-0.05	-0.35	-0.31	0.22	-0.59	-0.38	-0.21	-0.07
CAT	-0.65	0.80	-1.41	2.97	3.46	3.92	2.30	-2.86	0.32	-3.40	...	3.84	6.17	-5.90	-4.47	-1.54	3.30	-5.31	-3.46	-0.98	3.52
CSCO	0.52	0.27	-0.50	0.09	1.12	-3.41	0.01	-0.09	-0.22	-0.41	...	0.06	0.59	0.05	-0.65	-0.29	0.05	-0.58	-0.92	-0.17	-0.01
CVX	-0.47	1.88	0.84	-0.52	3.26	-0.83	3.22	2.87	1.47	-4.19	...	3.18	2.07	-6.60	-0.86	0.54	2.08	-3.61	-1.23	-0.87	-0.98
DD	-0.29	1.50	-1.18	1.85	2.38	1.96	1.54	-0.88	-0.35	-1.20	...	1.63	1.10	-1.79	-2.03	0.00	0.81	-2.89	-0.52	-0.46	2.46
DIS	1.71	0.28	0.67	-0.79	1.67	2.61	0.37	0.12	0.53	-0.60	...	1.30	0.76	-0.41	-1.80	0.24	0.37	-2.52	-0.68	-0.59	-0.27
GE	-0.06	0.21	0.76	0.27	0.43	0.56	-0.07	-0.06	-0.58	-0.04	...	0.18	0.31	-0.69	-0.08	-0.23	0.12	-0.86	-0.39	0.18	-0.37
HD	-0.82	1.73	0.54	0.24	-0.33	0.68	1.01	-2.22	0.07	-0.12	...	0.10	-0.46	-0.45	0.08	0.36	0.00	-1.55	-1.06	1.07	0.79
HPQ	2.87	1.39	1.09	-1.18	2.02	0.93	0.22	-4.89	-0.45	-1.17	...	1.03	-0.69	0.12	-0.38	-4.22	1.38	-1.17	-0.74	-0.13	0.11
IBM	0.72	3.00	5.68	3.79	4.82	-0.23	0.66	-1.29	-0.53	0.83	...	3.64	2.93	-3.22	1.53	0.35	-1.00	-3.39	-1.58	0.00	1.37

Datajoukkoa tarkastelemalla havaitaan nopeasti tarve datajoukon jonkinlaiselle skaalaukselle tai normalisoinnille. Osakkeiden viikoittainen hintakehitys nimittäin riippuu huomattavasti alkuperäisestä osakkeen arvosta ja kalliimpien osakkeiden tapauksessa hinta vaihtelee enemmän vaikkei suhteellinen muutos olisikaan yhtä suurta hinnaltaan halvempaan osakkeeseen verrattuna. Datalle käytetään nyt yksikkövektorinormalisointia, sillä se suoritetaan datalle sarakkeiden sijaan riveittäin. Tämä siitä syystä, että tavoitteena on yhdenmukaistaa osakkeiden hintakehityksen arvoalueet.

Kuvassa 10 on visualisoitu normalisoitua dataa kahdessa ulottuvuudessa. Datapisteet näyttävät erottuvan toisistaan hieman heikosti MDS ja t-SNE menetelmillä, mutta PCA:n tulosta tarkasteltaessa datasta on erotettavissa noin viisi tai kuusi rypästä. Myös UMAP onnistuu erottelemaan dataa selkeästi muutamaan klusteriin. Menetelmien vertailun alla on esitetty myös eräs K-means klusteroinnin

tulos PCA:n avulla dimensioltaan pudotetulle datalle, kun klustereiden määräksi asetettiin 6. Klustereiden havaitsemisen jälkeen tehtävänä olisi vielä saatujen tulosten mielekkyyden arviointi, mutta tähän ei tässä tutkielmassa paneuduta tarkemmin.

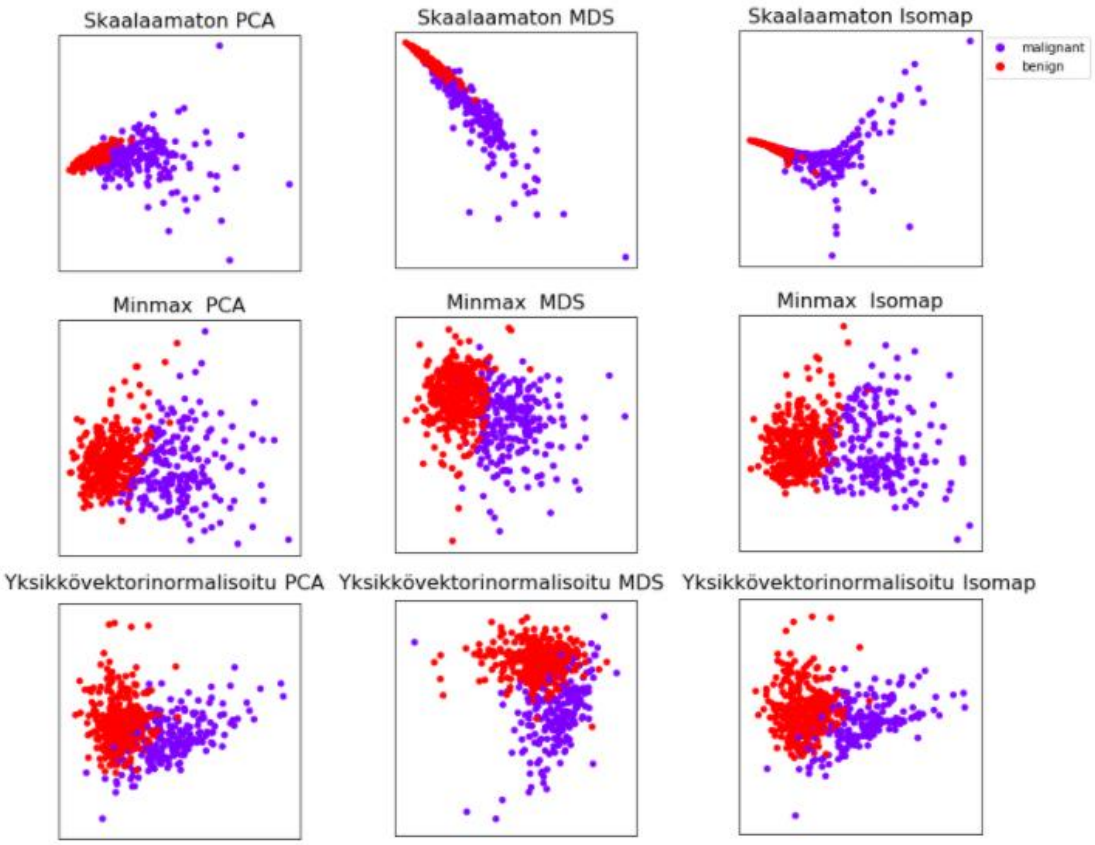


Kuva 10. Osakedatan visualisointi ja K-means klusterointi.

4.2.3. Rintasyöpädata

Vertaillaan vielä lopuksi, kuinka datan skaalaaminen ja normalisointi vaikuttaa dimensioiden vähentämisen tulokseen. Esimerkissä on käytetty kaksi eri luokkaa sisältävää rintasyöpädataa [37]. Datajoukossa on 569 näytettä ja 32 piirrettä.

Kuvassa 11 ylimmällä rivillä on esitetty skaalaamaton data, joka on pudotettu kahteen dimensioon kolmella eri dimensioiden vähennysmenetelmällä. Seuraavalla rivillä data on taas skaalattu nollan ja yhden välille ja alimmalla rivillä data on normalisoitu yksikkövektorinormalisoinnilla. Tuloksista käy ilmi, kuinka skaalaamattomalla datalla kaksi eri luokkaa eivät erotu kovinkaan hyvin toisistaan käytetyillä dimensioiden vähennysmenetelmillä. Skaalatun ja normalisoidun datan tapauksessa algoritmit taas tuottavat kaksidimensioisen esityksen datasta, jossa luokat ovat selkeämmin erillään.



Kuva 11. Rintasyöpädatan visualisointi eri menetelmillä.

5. YHTEENVETO

Tässä työssä on perehdytty data-aineiston esikäsittelyyn datan muokkaamisesta, sekä normalisoinneista ja skaalauksista dimensioiden pudotukseen ja datan visualisointiin. Näkökulmana on koneoppimisen tarpeet. Menetelmien matemaattista taustaa ja toimintaperiaatteita on avattu niiden erojen valaisemiseksi. Näiden seikkojen hahmottamisen tukemiseksi kehitettiin Python-kielinen työkalu, joka mahdollistaa erilaisten data-aineistojen käsittelyn ja visualisoinnin yleisimmillä menetelmillä.

Työn perimmäisenä tarkoituksena on tarjota vasta aihepiiriin perehtyvälle menetelmiä, joiden avulla koneoppimisen tuloksia on mahdollista parantaa ja auttaa myös soveltajaa hahmottamaan käsillä olevaa data-aineistoa. Tämän takia eri menetelmien vaikutusta havainnollistettiin todellisten esimerkkien avulla. Samalla demonstrointiin visualisoinnin tarpeellisuutta uuden datan kanssa työskenneltäessä.

Sopivan skaalauksen tai normalisoinnin, sekä dimensioiden vähentämismenetelmän valinta on riippuvaista datasta ja usein paras vaihtoehto on kokeilla useampien menetelmien toimivuutta. Dimensioiden vähentämisen kannalta on lisäksi tärkeää, että käytettävä menetelmä säilyttää alkuperäisen datan rakenteen ja riippuvuussuhteet myös matalammassa ulottuvuudessa.

6. LÄHTEET

[1] Hotelling H. (1933) Analysis of a complex of statistical variables into principal components. Journal of Educational Psychology, 24:417–441,498–520.

[2] Torgerson W. S. (1952) Multidimensional scaling I: Theory and method. Psychometrika, 17:401–419.

[3] Tenebaum J. B., Silva V., Langford J. C. (2000) A Global Geometric Framework for Nonlinear Dimensionality Reduction, Science 290, 2319-2323.

[4] Van Der Maaten L., Hinton G. (2008) Visualizing data using t-SNE. Journal of machine learning research 9 Nov, 2579–2605.

[5] McInnes L., Healy J., Melville J. (2018) Umap: Uniform manifold approximation and projection for dimension reduction. DOI: https://doi.org/10.21105/joss.00861.

[6] Saul L. K. (2020) A tractable latent variable model for nonlinear dimensionality reduction. PNAS, 117 (27) 15403-15408. DOI: https://doi.org/10.1073/pnas.1916012117.

[7] Github repository: https://github.com/markmult/Data-visualizer.

[8] Salama M. A., Hassanien A. E., Fahmy A. A. (2010) Reducing the influence of normalization on data classification. In Proceedings of International Conference on Computer Information Systems.

[9] Tax D. M. J. (2001) One-class classification, PhD Thesis, Delft University of Technology, http://www.ph.tn.tudelft.nl/˜davidt/thesis.pdf ISBN: 90-75691-05-x.

[10] Patel V. R., Metha R. G. (2011) Impact of outlier removal and normalization approach in modified k-means clustering algorithm. International Journal of Computer Science Issues, 8, 331-336.

[11] Sola J., Sevilla J. (1997) Importance of input data normalization for the application of neural networks to complex industrial problems, IEEE Transactions on Nuclear Science, Vol. 44, No: 3, 1464 – 1468.

[12] Jayalakshmi T., Santhakumaran A. (2011) Statistical Normalization and Back Propagation for Classification. International Journal of Computer Theory and Engineering, Vol. 3, No.1, 1793-8201.

[13] Aksoy S., Haralick R. (2001) Feature normalization and likelihood-based similarity measures for image retrieval. Pattern Recognit. Lett., Special Issue on Image and Video Retrieval.

[14] Nayak S. C, Misra B. B., Behera H. S. (2014) Impact of Data Normalization on Stock Index Forecasting, International Journal of Computer Information Systems and Industrial Management Applications, Vol. 6, 257 – 269.

[15] Singh B. K., Verma K., Thoke A. S. (2015) Investigations on Impact of Feature Normalization Techniques on Classifier's Performance in Breast Tumor Classification, International Journal of Computer Applications Vol. 116, No. 19.

[16] Albon C., (2018) Machine Learning with Python Cookbook, O'Reilly Media Inc.

[17] Cleveland W.S., McGill R. (1984) The many faces of a scatterplot. Journal of the American Statistical Association, 79:807–822.

[18] Fisher R.A. (1936) The use of multiple measurements in taxonomic problems. Annual Eugenics, 7(Part II):179–188, 179-188.

[19] Lee J. A., Verleysen M. (2007) Nonlinear Dimensionality Reduction, Springer Science & Business Media.

[20] Huang G. B., Ramesh M., Berg T., Learned-Miller E. (2007) Labeled Faces in the Wild: A Database for Studying Face Recognition in Unconstrained Environments. University of Massachusetts, Amherst, Technical Report 07-49.

[21] Zhu C., Idemudia C. U., Feng W. (2019) Improved logistic regression model for diabetes prediction by integrating PCA and K-means techniques, Informat. Med. Unlocked, vol. 17.

[22] Kaya I. E., Pehlivanlı A. Ç., Sekizkardeş E. G., Ibrikci T. (2017) PCA based clustering for brain tumor segmentation of T1w MRI images, Comput. Methods Programs Biomed., vol. 140, 19-28.

[23] G. T. Reddy (2020) Analysis of Dimensionality Reduction Techniques on Big Data, IEEE Access, vol. 8, 54776-54788.

[24] Duda R. O., Hart P. E., Stork D. G. (2012) Pattern classification. John Wiley & Sons.

[25] Gower J. C. (1966) Some distance properties of latent root and vector methods used in multivariate analysis. Biometrika, 53:325–338.

[26] Kruskal J. B. (1964). Multidimensional scaling by optimizing goodness of fit to a nonmetric hypothesis. Psychometrica, 29:1–26.

[27] Sammon Jr. J. W. (1969) A nonlinear mapping for data structure analysis. IEEE Transactions on Computers 18, 401–409.

[28] Gashi I., Stankovic V., Leita C., Thonnard, O. (2009) An Experimental Study of Diversity with Off-the-shelf AntiVirus Engines, Proceedings of

The Eighth IEEE International Symposium on Networking Computing and Applications, NCA 2009, July 9-11.

[29] Jamieson A. R., Giger M. L., Drukker K., Li H., Yuan Y., Bhooshan N. (2009) Exploring nonlinear feature space dimension reduction and data representation in breast CADx with Laplacian eigenmaps and t-SNE, Medical Physics, Vol 37, No. 1, 339-351.

[30] Van Der Maaten L. (2014) Accelerating t-SNE using tree-based algorithms. The Journal of Machine Learning Research 15(1), 3221–3245.

[31] Becht E., McInnes L., Healy J., Dutertre C. A., Kwok I. W., Ng L. G., Newell E. W. (2019). Dimensionality reduction for visualizing single-cell data using UMAP. Nature biotechnology, *37*(1), 38-44.

[32] Venna, J. (2007) Dimensionalisuuden pienentäminen samankaltaisuuksien visuaalista tarkastelua varten. Väitioskirja, Teknillinen korkeakoulu, Dissertations in Computer and Information Science.

[33] Vipunen – opetushallinnon tilastopalvelu: Yliopistojen tutkinto-opiskelijoiden kansainvälinen liikkuvuus. Opetushallinnon ja Tilastokeskuksen tietopalvelusopimuksen aineisto 2.10. https://vipunen.fi/fi-fi/_layouts/15/xlviewer.aspx?id=/fi-fi/Raportit/Yliopistojen%20tutkinto-opiskelijoiden%20kansainv%C3%A4linen%20liikkuvuus%20-%20yliopisto.xlsb, luettu 10.8.2020.

[34] Graham J.W. (2009) Missing data analysis: making it work in the real world. Annual Review of Psychology, 60: 549–576.

[35] Ilmatieteenlaitos avoin data, https://www.ilmatieteenlaitos.fi/havaintojen-lataus, luettu 4.8.2020.

[36] Brown, M. S., Pelosi, M. & Dirska, H. (2013) Dynamic-radius Species-conserving Genetic Algorithm for the Financial Forecasting of Dow Jones Index Stocks. Machine Learning and Data Mining in Pattern Recognition, 7988, 27-4. https://archive.ics.uci.edu/ml/datasets/Dow+Jones+Index.

[37] Wolberg W. H., Street W. N.n Mangasarian O. L. (1995), https://archive.ics.uci.edu/ml/datasets/Breast+Cancer+Wisconsin+(Diagnostic), luettu 11.8.2020.